

DEEProtect: Enabling Inference-based Access Control on Mobile Sensing Applications

Changchang Liu¹ Supriyo Chakraborty² Prateek Mittal¹
¹ Princeton University, ² IBM T.J. Watson Research Center

Abstract

Personal sensory data is used by context-aware mobile applications to provide utility. However, the same data can be used by an adversary to make sensitive inferences about a user thereby violating her privacy.

We present DEEProtect, a framework that enables a novel form of access control that we refer to as the *inference-based access control*, in which mobile apps with access to sensor data are limited (provably) in their ability to make inferences about user’s sensitive data and behavior. DEEProtect adopts a two-layered privacy strategy. First, it leverages novel deep learning techniques to perform data minimization and limits the amount of information being shared; the learning network is used to derive a compact representation of sensor data consisting only of features relevant to authorized utility-providing inferences. Second, DEEProtect obfuscates the previously learnt features, thereby providing an additional layer of protection against sensitive inferences; our approach can provide both conventional and relaxed notions of local differential privacy, depending on how sensitive inferences are specified. Through theoretical analysis and extensive experiments using real-world apps and datasets, we demonstrate that when compared to existing approaches DEEProtect provides provable privacy guarantees with up to 8x improvement in utility. Finally, DEEProtect shares obfuscated but raw sensor data reconstructed from the perturbed features, thus requiring no changes to the existing app interfaces.

1 Introduction

Mobile devices such as smartphones and wearable devices are increasingly gaining popularity as platforms for collecting and sharing sensor data. These devices, which are often equipped with sensors such as accelerometer, orientation sensor, magnetometer, camera, microphone, GPS and so on, are being used by mobile sensing systems to make sophisticated inferences about users. These

inferences have in turn enabled an entire ecosystem of context-aware apps such as behavior-based authentication [18, 31, 46, 56, 73, 87], fitness monitoring based on activity tracking [5, 55, 71], speech translation [47, 59] and traffic/environmental monitoring [3, 61, 77, 80].

While shared data has enabled context-aware apps to provide utility to the users, the same data can also be used by an adversary to make inferences that are possibly sensitive to the user such as speaker identity [50, 63], keystroke detection [13, 52, 57, 85], location tracking [12, 43, 63], device placement [65], onscreen taps recognition [60], onset of stress [16, 53] and detection of emotional state [16, 69]. Therefore, there exist fundamentally conflicting requirements between protecting privacy of the sensitive information contained in mobile sensor data and preserving utility of the same data for authorized inferences. We consider the following case studies that further illustrate this privacy-utility tradeoff:

- Case Study 1: A user shares accelerometer data with an authentication app that performs keyless validation of user identity (*useful inference*) [31, 46, 56]. But the same data can also be used to infer the activity mode of the user – if she is walking, or standing still, or moving up or down the stairs (*sensitive inferences*) [5, 55, 71] – which in turn may lead to more serious inferences regarding living habits and health conditions.
- Case Study 2: A user shares location and accelerometer data with a real-time life-logging app that helps with travel planning (*useful inferences*) [34, 43]. However, the same data can also be maliciously used to extract sequences of the user’s entered text (e.g., Passwords or PINs) (*sensitive inference*) [13, 57, 60, 85].
- Case Study 3: A user shares speech data with a voice-based search app that translates speech to text for web searches (*useful inference*) [47, 59]. But the same data can also be used to recognize the user’s identity (*sensitive inference*) [50, 63], compromising her privacy.

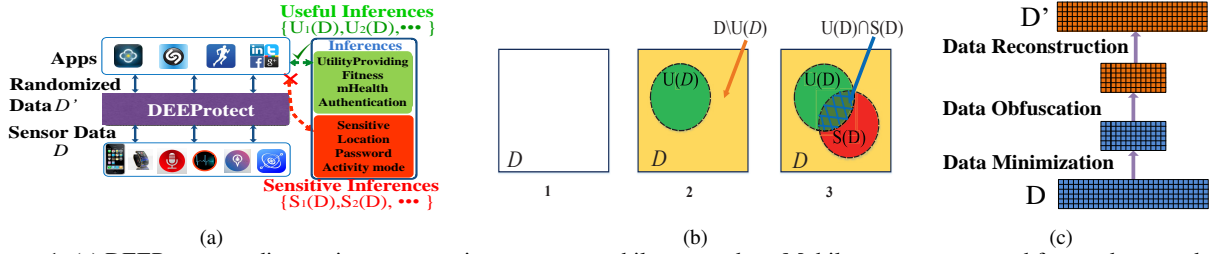


Figure 1. (a) DEEProtect mediates privacy-preserving access to mobile sensor data. Mobile apps can access obfuscated sensor data for providing utility, but are provably limited in their ability to infer sensitive information. (b) Venn diagram for information measures associated with user’s raw sensor data D (shown in white), useful inferences $U(D)$ (shown in green) and sensitive inferences $S(D)$ (shown in red). (c) DEEProtect pipeline comprises of a data minimization step for learning a compact feature representation specific to authorized useful inferences; a data obfuscation step to perturb features and provide provable privacy guarantees; a data reconstruction step to generate obfuscated sensor data from perturbed features.

Recently, several sensor privacy protection mechanisms (SPPMs) have been proposed [4, 7, 15, 19, 21, 22, 33, 39, 48, 74]. However, most existing SPPMs only provide binary access control over the sensor data that requires users to choose between sharing or blocking a sensor – limiting their applicability and adoption [7, 33, 39, 74]. A limited set of SPPMs that do focus on obfuscating sensor data, however, lack provable privacy guarantees [4, 15].

In this paper, we propose a new framework called DEEProtect that mediates privacy-preserving access to mobile sensor data. DEEProtect enables a new permission model where instead of blocking or allowing sensors, users specify their utility and privacy preferences in terms of inferences that can be derived from mobile sensor data. The overall information flow of DEEProtect is summarized in Figure 1(a).

In DEEProtect, users can specify as input a set of useful/authorized inferences denoted by $U(D) = \{U_1(D), U_2(D), \dots\}$ (e.g., behavior-based authentication, speech-to-text translation). All other inferences (possibly unknown) are considered sensitive by default. Alternatively, a subset of sensitive inferences can be specified by the users, which is denoted by $S(D) = \{S_1(D), S_2(D), \dots\}$ (e.g., detection of the text entered in keyboard, speaker identity recognition). These useful and sensitive inferences represent functions that map the mobile sensor data to their corresponding inference results (i.e., labels), which is the information we want to share (for useful inferences) or protect (for sensitive inferences). Outgoing sensor data is initially intercepted by DEEProtect, which obfuscates the data before sharing such that $U(D)$ can be derived accurately by the app and $S(D)$ is kept private.

Approach Overview: We now illustrate the working of DEEProtect by using Case Study 3 as an example. Figure 1(b) shows the venn diagram of information measures in DEEProtect. To protect users’ privacy, it is required that we should not directly share the raw sensor data D (the entire white square shown in Figure 1(b)–

1) with the untrusted third-party apps. In Case Study 3, D corresponds to the microphone data. Our objective is to enable $U(D)$ (speech-to-text translation) with high accuracy while protecting against $S(D)$ (speaker identity recognition). The key idea in DEEProtect is to first ignore the information that is orthogonal to authorized (useful) inference information. In this case, we only retain information that enable $U(D)$ and ignore the remaining information $U^c(D) = D \setminus U(D)$ (shown in yellow in Figure 1(b)–2). In the next step, we perturb those information that are correlated to both the useful and sensitive inferences, $U(D) \cap S(D)$ (shown in blue in Figure 1(b)–3), to satisfy provable privacy guarantees. This helps in selectively hindering the sensitive inferences without severely degrading the useful inferences.

For arbitrary useful and sensitive inferences, the three-stage data obfuscation pipeline used in DEEProtect is illustrated in Figure 1(c). In the first stage, we extract features from raw sensor data and perform *data minimization* [14, 40, 66] to guarantee that only features that are relevant to the authorized (useful) inferences are retained for further processing. Specifically, DEEProtect modifies an autoencoder network [37, 45, 82] to *automatically* explore the inherent structural characteristics of the sensor data, and learn a sparse feature representation of the data specific to the useful inferences. At this point, some of the extracted features, although highly specific to the useful inferences, might still have some correlation with the sensitive inferences. In the next step of the pipeline, to provide provable privacy guarantees for a single user’s mobile sensor stream, we exploit *local differential privacy* [23, 30, 68] to obfuscate the extracted features. Note that differential privacy [24–27, 29] is typically applied in multi-user settings, and local differential privacy can be applied in single-user settings. Specifically, we develop a computationally efficient mechanism that perturbs the sparse features, using either the conventional local differential privacy framework [23, 30, 68] or our relaxed notion of local differential privacy framework, depending

on users’ specification of sensitive inferences. Finally, we reconstruct obfuscated sensor data from the perturbed features such that the existing interfaces for mobile apps can continue to work without any change.

Usage Mode: DEEProtect supports two usage modes, depending on how sensitive inferences are specified.

Usage Mode 1: Only the useful inference set $U(D)$ is provided and the sensitive inference set $S(D)$ is the default set of ‘all possible inferences’. Under this mode, the user is interested in rigorous privacy guarantees with respect to the entire set (possibly unknown) of inferences. DEEProtect first implements deep learning based feature extraction method to raw sensor data and then applies local differential privacy to these extracted features.

Usage Mode 2: Both the useful and the sensitive inference sets are provided. Under this scenario, the user is interested in protecting only a specific set of sensitive inferences (and not all possible inferences). To provide provable privacy guarantees, we develop a relaxed variant of local differential privacy and use it to perturb the extracted features in a fine-grained manner.

Our theoretical analysis (Section 6) and experimental results on real-world datasets (Section 8) show that both usage modes of our approach significantly outperform previous state-of-the-art with up to 8x improvement.

Contributions: We now summarize our contributions under three main thrusts:

- *Provable Privacy Guarantees in Single-user Settings:* We support rigorous local differential privacy guarantees for a single user’s mobile sensor data. However, this rigorous privacy guarantee can require significant addition of noise impacting application utility. Thus, we propose a novel relaxed variant of local differential privacy for providing inference-based access control over the mobile sensor data. Our proposed privacy metric satisfies composition properties, and can achieve improved utility by focusing on protecting a specific set of inferences.
- *Novel Perturbation Mechanisms for Enhanced Privacy-Utility Tradeoffs:* We propose an effective perturbation mechanism which consists of two key techniques: deep learning based data minimization and feature obfuscation based data perturbation. Our first technique automatically extracts features relevant to the useful/authorized inferences by extending the deep learning models for dimensionality reduction, which to our best knowledge is the first such attempt. For the second technique, we propose effective feature perturbation methods, that achieve both conventional and relaxed notions of local differential privacy guarantees. Through rigorous theoretical analysis, we demonstrate the advantage of our approach over the state-of-the-art obfuscation mechanisms.

- *Implementation and Evaluation:* We implement our system using real-world datasets and mobile apps (for all the case studies discussed previously). For our first feature learning step, we demonstrate that our modified deep learning network outperforms the state-of-the-art feature extraction approaches with up to 2x improvement. For our end-to-end approach combining both feature extraction and perturbation, we demonstrate the advantage of our technique over the state-of-the-art obfuscation mechanisms with up to 8x improvement. We also implement experiments to show the effectiveness of our method in defending against inference attacks leveraging mobile sensor data. We will provide an open source software framework for all the experiments carried out in our study for independent verification and extensions of our results.

2 Related Work

2.1 Android-based SPPMs

MockDroid [7] is a modified version of the Android operating system (OS) with ability to ‘mock’ a given resource requested by an app. This approach allows users to revoke access to particular resources at run-time, encouraging users to consider the trade-off between functionality and privacy. AppFence [39] offers two approaches for protecting sensitive data from untrusted apps: shadowing and blocking sensitive data from being exfiltrated off the device. IdentiDroid [74] is also a customized Android OS providing different identifications and privileges, to limit the uniqueness of device and user information. The above systems still rely on the user to determine the sensitive sensors and block, mock, or shadow them. ipShield [15], provides users the ability to specify their privacy preferences in terms of inferences, however, they only generate binary privacy policies of *allow* and *deny* for individual sensors. Such binary access control over the sensor data may seriously affect the functionality of various apps. DEEProtect on the other hand provides the much needed automatic translation from higher-level privacy specifications in terms of inferences to obfuscation of sensor data.

Another important limitation of the state-of-the-art SPPMs is that they are often heuristic in nature and fail to provide rigorous privacy guarantees for inference-based access control over sensors in mobile devices. For instance, ipShield [15] does provide users with ability to configure obfuscation policies for raw sensor data (through addition of random noise), but does not quantify any privacy guarantees for those policies. Boxify [4] provides app sandboxing for untrusted apps and can be used to run an untrusted app in an isolated environment with minimum access privileges. EaseDroid [83] uses semi-

supervised learning to perform automatic policy analysis and refinement. This could be used to compliment DEEProtect by automatically learning the set of inferences that need to be protected and ones that need to be released from access patterns and audit logs.

Privacy-preserving mobile data aggregation has been studied in [19, 21, 48]. However, they assume the presence of a multi-user database whereas we aim to protect inference-based privacy for a single user’s sensor stream.

2.2 Repository of Inferences

We surveyed more than 70 research papers published in relevant conferences and journals over the past 6 years to form a knowledge repository of inferences (see Table 1 in the Appendix) that can be made using a combination of sensors accessed by mobile apps. This table forms the universe of possible inferences over which the set of useful and sensitive inferences in DEEProtect are defined.

2.3 Summary of Our Novelty

To the best of our knowledge, DEEProtect is the first system that provides a novel inference-based access control in mobile sensing applications. Furthermore, DEEProtect 1) provides provable privacy guarantees including both conventional local differential privacy and our relaxed variant of local differential privacy for sensitive sensor data; 2) presents an effective mechanism consisting of two key techniques: deep learning based data minimization and feature obfuscation based data perturbation; 3) outperforms the state-of-the-art methods in multiple mobile sensing datasets and applications.

3 Threat Model and System Model

We assume that DEEProtect together with the underlying mobile OS, sensors, and system services form the trusted domain. Our adversary is an untrusted app provider, who publishes apps in the marketplace for advertised useful inferences. The app accesses sensors including ones to which the user has provided explicit permission, and also others for which no permission is required. The app would then send out the collected sensor data to the adversary. Our goal is to ensure that the data shared with the app can only be used for deriving the authorized (useful) inferences and not any sensitive inferences.

We aim to limit/bound the adversary’s inference of the user’s sensitive information via access to the obfuscated sensor data while achieving two rigorous privacy properties: (a) first, we aim to provide rigorous local differential privacy guarantees [23, 30, 68], that limit an adversary against all possible inferences (corresponding to usage mode 1 in Section 1); and (b) second, we aim to provide

a novel relaxed variant of local differential privacy that limits an adversary against a specified set of sensitive inferences (corresponding to usage mode 2 in Section 1). Furthermore, we adapt the popular Laplace perturbation mechanism [24] to construct effective perturbation approaches, in order to provide both conventional and relaxed notions of local differential privacy guarantees for protecting mobile sensor data (in Sections 5.2.2, 5.2.3).

It is also interesting to note that a threat model similar to usage mode 2 (i.e., protecting against specified inferences) has been explored in Pufferfish privacy [42] and Blowfish privacy [35]. For example, the set of sensitive inferences computed over sensor data in DEEProtect is the set of potential secrets defined in Pufferfish and Blowfish privacy. Blowfish privacy has recently been adopted by the U.S. Census Bureau demonstrating its usefulness as an enabler for practical deployment.

4 Privacy Metrics for Protecting Mobile Sensor Data in Single-user Settings

4.1 Local Differential Privacy

Differential privacy (DP) [24–27, 29] is a rigorous mathematical framework that prevents an attacker from inferring the presence or absence of a particular record in a statistical database. DP randomizes the query results, computed over the *multi-user* database, to ensure that the risk to an individual record’s privacy does not increase substantially (bounded by a function of the privacy budget ϵ) as a result of participating in the database. Local differential privacy (LDP) [23, 30, 68], as an adaption of DP, is defined under the setting where the user does not trust anyone (not even the central data collector). Local privacy dates back to Warner [84], who proposed the randomized response method to provide plausible deniability for individuals responding to sensitive surveys. In our setting, we therefore apply LDP to protect a *single* user’s mobile sensor data while satisfying rigorous privacy guarantees. Similar applications have also been explored in Google’s Rappor System [30]. Specifically, ϵ -LDP is defined as follows:

Definition 1 (ϵ -LDP) [23] A randomized algorithm $A(\cdot)$ provides ϵ -LDP if for any two databases D_1, D_2 and for any output set o ,

$$\max_{D_1, D_2, o} \frac{\mathbb{P}(A(D_1) = o)}{\mathbb{P}(A(D_2) = o)} \leq \exp(\epsilon) \quad (1)$$

where $A(D_1)$ (resp. $A(D_2)$) is the output of $A(\cdot)$ on input D_1 (resp. D_2) and ϵ is the privacy budget. Smaller value of ϵ corresponds to a higher privacy level.

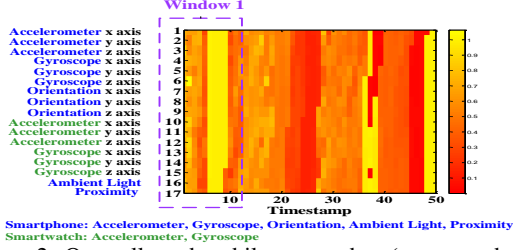


Figure 2. Our collected mobile sensor data (segmented with a time window size $N_w = 10$).

4.1.1 Applying LDP to Mobile Sensor Data

For a mobile sensing system, consisting of N_s sensors across T timestamps, the temporal mobile sensor matrix $\mathbf{TMS} \in \mathbb{R}^{N_s \times T}$ is a real matrix where $\mathbf{TMS}(i, j)$ records the sensing data of the i -th sensor at the j -th timestamp. To fully explore the temporal characteristics of the data, we follow common practice [31, 46, 54] and partition it into a series of segments according to a user-specified time window size N_w as shown in Figure 2. For the w -th window, we stack the corresponding columns within it to form a column vector which is denoted by $\mathbf{x}_t \in \mathbb{R}^{N_s N_w \times 1}$. The temporal mobile sensor matrix can thus be reformulated as $\mathbf{X} = [\mathbf{x}_1; \mathbf{x}_2; \dots]$, where $\mathbf{X} \in \mathbb{R}^{N_s N_w \times \frac{T}{N_w}}$. We therefore apply LDP to a single user’s mobile sensor data which is segmented according to a user-specified window size. In this way, we can take the temporal dynamics of users’ mobile sensor data into consideration, while providing rigorous privacy guarantees.

From Definition 1, we observe that ϵ -LDP has the same mathematical formulation as ϵ -DP, except that the neighboring databases in LDP are *any two possible databases without the constraint of one tuple’s difference as in DP* [24]. Thus, when apply LDP to mobile sensing applications, the neighboring databases represent any two segmented sensor data that differ in *any possible sensor recording at any timestamp within the same window*. A mechanism that provides rigorous LDP can defend against any inference attacks over the obfuscated data, according to the post-processing invariant property [28]. Therefore, we can provide rigorous LDP for users’ sensor data to defend against all possible inferences over the data (corresponding to usage mode 1 in Section 1).

4.2 Relaxed Local Differential Privacy

The need for a relaxed variant: Although we can provide rigorous LDP and defend against all inferences computed over the obfuscated sensor data (in usage mode 1), in practice, it may suffice to protect a specific subset of sensitive inferences instead of focusing on all possible inferences. Therefore, we further consider a scenario where the user aims to defend against a *specific*

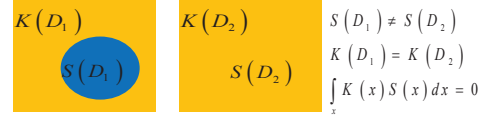


Figure 3. Relaxed neighboring databases according to Definition 2, which have different sensitive inference values whilst the same values for the orthogonal inferences.

subset of sensitive inferences over the sensor data (corresponding to usage mode 2). Under this threat model, there are novel opportunities to strategically add noise and enhance utility. We thus relax the definition of LDP to provide provable privacy guarantees for a specific subset of sensitive inferences but significantly improve the utility performance (will discuss in Sections 6, 8).

Pufferfish [42] and Blowfish [35] privacy frameworks have similarly motivated the threat model of protecting against a specific set of sensitive computations over the data and have been deployed in real-world settings. Inspired by these frameworks and Definition 1, we define our relaxed neighboring databases as follows:

Definition 2 *Let us represent a set of sensitive inferences as $S(\cdot)$, and let us represent an orthogonal function of $S(\cdot)$ as $K(\cdot)$. Thus, $\int_x K(x) \times S(x) dx = 0$, where $K(\cdot) \in S^\perp$ and S^\perp is the orthogonal component of $S(\cdot)$ consisting of all the functions that are orthogonal to $S(\cdot)$. Two databases D_1, D_2 are relaxed neighboring databases, iff. $S(D_1) \neq S(D_2)$, $K(D_1) = K(D_2)$ for any function $K(\cdot)$.*

We illustrate the relaxed neighboring databases above in Figure 3. For concrete interpretation of the definition, we refer back to Case Study 3 in Section 1, where the useful inference is *speech-to-text translation* and the sensitive inference is *speaker identity recognition*. Intuitively, both the useful and sensitive inferences rely on a limited number of features extracted from the sensor data. The commonly used features for speech-to-text translation are Mel-frequency Cepstral Coefficients (MFCC), spectrogram and N-gram [47, 59, 86], while the popular features for speaker identity recognition are MFCC, spectrogram and the voice biometrics of the user (such as her pitch, accent, etc.) [50, 63]. According to Definition 2, the relaxed neighboring databases there refer to two sensor data streams that correspond to different speakers (i.e., different values of MFCC, spectrogram and the voice biometrics) and have the same values of N-gram (the orthogonal component). Therefore, the relaxed neighboring databases in Definition 2 focuses on protecting the privacy of sensitive inferences only (whose values differ in the two databases), and not the component of the data that is orthogonal to sensitive inferences (whose values remain the same in the two databases).

Remark on orthogonal functions: We further illustrate the relaxed neighboring databases in a three-dimensional

space (our definition can be generally applied to any space) in Figure 4, where there are two functions $K_1(\cdot), K_2(\cdot)$ that are orthogonal with the sensitive inference function $S(\cdot)$, i.e., $\int_x K_1(x) \times S(x) dx = 0, \int_x K_2(x) \times S(x) dx = 0$. According to Definition 2, we have the following properties for the two relaxed neighboring databases D_1, D_2 : 1) they have the same projection on the direction corresponding to the orthogonal functions, i.e., $K_1(D_1) = K_1(D_2), K_2(D_1) = K_2(D_2)$; 2) they have different projection on the direction corresponding to the sensitive inference function, i.e., $S(D_1) \neq S(D_2)$.

Inspired by the relaxed neighboring databases in Definition 2, we now formulate a relaxed variant of local differential privacy (RLDP) which is also an instantiation of the Pufferfish and Blowfish privacy frameworks. In fact, we can observe that our set of sensitive inferences computed over sensor data is the set of potential secrets defined in Pufferfish privacy (Definition 3.4 in [42]) and Blowfish privacy (Definition 4.2 in [35]).

Definition 3 (ϵ -RLDP) For a set of sensitive inference functions $S(\cdot)$ and a privacy budget ϵ , a randomized algorithm $A(\cdot)$ satisfies ϵ -RLDP for protecting $S(\cdot)$, if for any two relaxed neighboring databases D_1, D_2 defined in Definition 2 and any output set o ,

$$\max_{D_1, D_2, o} \frac{\mathbb{P}(A(D_1) = o)}{\mathbb{P}(A(D_2) = o)} \leq \exp(\epsilon) \quad (2)$$

Smaller values of ϵ correspond to higher privacy level.

To achieve ϵ -RLDP for protecting sensitive information, it is required that the shared data $o = A(D_1) = A(D_2)$ conditioned on any two different sensitive information $S(D_1), S(D_2)$ (where $S(D_1) \neq S(D_2)$ in Definition 2), is statistically indistinguishable from each other. This in turn guarantees that an adversary gains negligible information about the true sensitive information upon observing the shared data. We re-iterate that unlike traditional LDP frameworks, its relaxed variant focuses only on the privacy of a specific subset of sensitive inferences. A perturbation mechanism developed to satisfy RLDP guarantees would thus result in better utility performance than the LDP mechanisms (will discuss in Sections 6, 8).

By extending the previous results on composition properties for LDP in [23], our privacy guarantees also compose securely, i.e., retain privacy guarantees even in the presence of multiple independent releases (detailed proof is deferred to appendix to improve readability).

Theorem 1 (Sequential Composition Theorem) Let randomized algorithm A_t ($t = 1, 2, \dots$) each provide ϵ_t -RLDP under the sensitive inference functions $S(\cdot)$. The sequence of these algorithms $A_t(D)$ provides $\sum_t \epsilon_t$ -RLDP.

Theorem 2 (Parallel Composition Theorem) Let randomized algorithms A_t ($t = 1, 2, \dots$) provide ϵ_t -RLDP

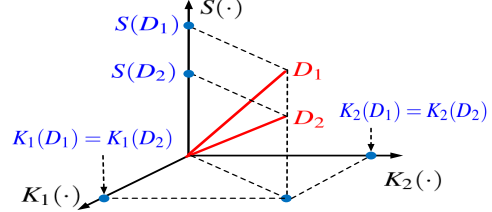


Figure 4. Illustration of the two relaxed neighboring databases D_1, D_2 in Definition 2. $K_1(\cdot), K_2(\cdot)$ are orthogonal functions of the sensitive inference function $S(\cdot)$.

under the sensitive inference functions $S(\cdot)$ and D_t be arbitrary disjoint data set. The sequence of these randomized algorithm $A_t(D_t)$ provides $\max_t \epsilon_t$ -RLDP.

5 Perturbation Mechanisms

In this section, we design effective perturbation mechanisms to achieve ϵ -LDP and ϵ -RLDP (corresponding to the two usage modes in Section 1) for protecting mobile sensor data in a single-user setting. Our key insight is to exploit the structural characteristics of mobile sensor data to enhance privacy-utility tradeoffs. Before proposing our privacy mechanisms, we first introduce a baseline approach which directly generalizes the traditional DP perturbation mechanisms to achieve ϵ -LDP.

5.1 Baseline Approach

To achieve ϵ -DP, the Laplace Perturbation Mechanism (LPM) [24] applies noise drawn from a suitable Laplace distribution to perturb the query results. More formally, for a query function $Q(\cdot)$, LPM computes and outputs $A(D) = Q(D) + \text{Lap}(\lambda)$, where $\lambda = \Delta Q / \epsilon$ is the parameter of the Laplacian noise and $\Delta Q = \max_{D_1, D_2} \|Q(D_1) - Q(D_2)\|_1$ is the global sensitivity of $Q(\cdot)$ [24].

When applying LDP (recall Definition 1) on segmented mobile sensor data \mathbf{x}_t , the neighboring databases $\mathbf{x}_{t1}, \mathbf{x}_{t2}$ may differ in all their possible tuples (i.e., all sensor recordings across all timestamps within the same window), while the neighboring databases in traditional DP frameworks only differ in one tuple. According to the *composition theorem* of DP [58], we propose a baseline approach to achieve ϵ -LDP, which inserts a Laplacian noise to each temporal sensor data point with the same parameter of $\lambda = \dim(\mathbf{x}_t) \Delta Q / \epsilon$ ¹, where $\dim(\mathbf{x}_t)$ is the dimension of each segmented sensor data \mathbf{x}_t (we refer interested readers to [17] for similar perturbation mechanisms). The baseline approach achieves ϵ -LDP guarantees and the corresponding proof is deferred to the appendix to improve readability.

¹Note that the sensitivity ΔQ is estimated from our collected data serving as a local sensitivity.

5.2 DEEProtect Perturbation Mechanisms for Enhanced Privacy-Utility Tradeoffs

Note that the baseline approach introduces noise that is linear in the number of temporal sensor recordings within the time window. This increases the magnitude of noise that needs to be added and thus degrades the usability of the data. Therefore, a better alternative to protect sensor data is to build a compact, privacy-preserving synopsis from the data by exploiting its structural characteristics.

Data minimization [14, 40, 66] is a fundamental legal instrument that protects privacy by limiting the collection of personal data to the minimum extent necessary for attaining legitimate goals. In our work, we enforce the principle of data minimization and retain only the minimum number of features necessary to enable the authorized inferences. These extracted features may still be correlated with sensitive inferences; thus we incorporate local differential privacy to obfuscate the extracted features for protecting against sensitive inferences.

In this section, we propose DEEProtect (recall its pipeline in Figure 1(c)) which consists of two key steps of 1) first extracting features based on the data minimization principle and 2) then perturbing these features to achieve both conventional and relaxed notions of LDP guarantees (corresponding to the two usage modes in Section 1 respectively). We will demonstrate the significant advantages of our mechanism over the existing privacy methods theoretically as well as using multiple real-world case studies on real datasets.

5.2.1 Deep Learning based Data Minimization

Mobile sensor data is usually high-dimensional in nature, which typically exhibits both structure and redundancy, allowing minimization [6, 49]. This lays the foundation for the first technique in our DEEProtect system: deep learning based data minimization.

Existing Autoencoder Models: Deep learning models [37, 45] learn multi-layer transformations from the input data to the output representations, which is more powerful for feature extraction than hand-crafted shallow models. Among the building blocks of these models, autoencoders [37] automatically extract features in an unsupervised manner by minimizing the reconstruction error between the input and its reconstructed output.

A single-layer autoencoder is shown in Figure 5. The encoder function $E_{nc}(\cdot)$ maps the input data \mathbf{x}_t to the hidden units (features) according to $\mathbf{f}_t = E_{nc}(\mathbf{x}_t) = g_{enc}(\mathbf{W}\mathbf{x}_t + \mathbf{b}_e)$, where $g(\cdot)$ is typically a sigmoid function, \mathbf{W} is a weight matrix and \mathbf{b}_e is a bias vector. The decoder function $D_{ec}(\cdot)$ maps these features back to the original input space according to $\tilde{\mathbf{x}}_t = D_{ec}(\mathbf{f}_t) = g_{dec}(\mathbf{W}'\mathbf{f}_t + \mathbf{b}_d)$, where $g_{dec}(\cdot)$ is usually the same form as that in the encoder, \mathbf{W}' is a weight matrix and \mathbf{b}_d a

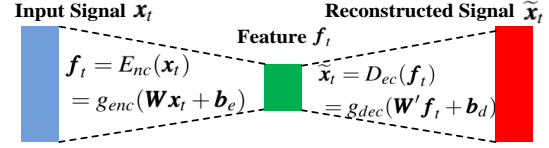


Figure 5. Illustration of Autoencoder Models.

bias vector. Considering the inherent structure of the mobile sensor data [6, 49], we aim to learn an effective feature space on which the mobile sensor data would have a succinct representation, and the corresponding objective function is as follows.

$$J_0(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}') = \sum_{t=1}^{T/N_w} Re_error(\mathbf{x}_t, D_{ec}(E_{nc}(\mathbf{x}_t))) \quad (3)$$

where $Re_error(\mathbf{x}_t, D_{ec}(E_{nc}(\mathbf{x}_t)))$ is the reconstruction error between the input data \mathbf{x}_t and its reconstructed output $D_{ec}(E_{nc}(\mathbf{x}_t))$ (detailed mathematical formulation for Re_error is deferred to the appendix to improve readability). Existing autoencoders thus aim to minimize $J_0(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}')$ with respect to $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}'$ [37, 45].

Constructing Data Minimization Model: Using a non-linear encoding function an autoencoder can typically extract better features than previous linear transformation methods [81]. But, the features learnt are not specific to the useful (authorized) inferences. To handle this, we explicitly modify the autoencoder models in Eq. 3 by incorporating the useful inferences and other associated constraints as follows.

Incorporating Useful Inferences: The objective for our data minimization mechanism is to maximize the utility performance with the minimum amount of information, therefore it is important to combine the useful inferences with the autoencoder models to automatically extract features in a supervised manner. *To the best of our knowledge, this is the first work to modify the deep learning models through incorporating the authorized inferences.* Specifically, we incorporate the minimization of cost function corresponding to the useful inferences to the objective function of existing autoencoders in Eq. 3.

We analyze the cost function for each inference using Case Study 1 in Section 1, where behavior-based authentication was considered as a useful inference and activity mode detection was deemed sensitive. Both the useful and sensitive inferences can be mathematically transformed into a classification problem, which can be addressed by machine learning techniques. For instance, by leveraging the popular *ridge regression* technique [38], we can learn an optimal classifier as follows.

$$\begin{aligned} \mathbf{c}^{*U/S} &= \arg \min_{\mathbf{c}} \mathbb{C}^{*U/S} \\ &= \arg \min_{\mathbf{c}} \beta \|\mathbf{c}\|^2 + \sum_{t=1}^{T/N_w} (\mathbf{c}^T \mathbf{x}_t - y_t^{U/S})^2 \end{aligned} \quad (4)$$

where $\mathbb{C}^{*U/S}$ represents the cost function for the useful and sensitive inferences respectively. For behavior-based authentication (*useful inference*), the label $y_t^U \in \{1, -1\}$ where 1 represents the legitimate user and -1 represents the adversary². Similarly, for activity mode detection (*sensitive inference*), $y_t^S \in \{1, 2, 3\}$ where the labels 1, 2 and 3 represent *walking*, *standing still* and *moving up or down the stairs* respectively. The optimal classifier $\mathbf{c}^{*U/S}$ learned in Eq. 4 can be utilized to label the newly-coming mobile sensor data for behavior-based authentication or activity mode detection. The cost function in Eq. 4 has been popularly used in machine learning community whilst other general models in [8, 79] can also be explored as potential cost function to reflect the prediction accuracy of the inferences. Note that our analysis are not restricted to these settings and can be utilized for arbitrary inference-based mobile applications.

Incorporating Orthogonality: In addition, we also aim to learn orthogonal features so that we can deal with each feature independently in our feature perturbation mechanism for achieving enhanced utility performance. Therefore, we add another constraint as $\mathbf{W}\mathbf{W}^T = \mathbf{I}$ to the objective function to guarantee the *orthogonality* of the features (similar intuition has also been utilized in [78]).

Our New Autoencoder Model: Incorporating the two constraints (useful inferences and orthogonality) into Eq. 3 and combining with Eq. 4, we construct the objective function for our data minimization method as

$$\begin{aligned} \min J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c}) = \min J_0(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}') \\ + \mu\beta \|\mathbf{c}\|^2 + \mu \sum_{t=1}^{T/N_w} (\mathbf{c}^T (D_{ec}(E_{nc}(\mathbf{x}_t)) - y_t^U)^2) \quad (5) \\ \text{s.t. } \mathbf{W}\mathbf{W}^T = \mathbf{I} \end{aligned}$$

where μ controls the trade-off between the reconstruction loss and the utility penalty, and $\mathbf{W}\mathbf{W}^T = \mathbf{I}$ represents the orthogonality constraint. Note that the last two terms in Eq. 5 correspond to the cost function in ridge regression (recall Eq. 4). Although we only consider the cost function of ridge regression to optimize the data minimization process, our algorithm can be generalized to multiple machine learning techniques such as support vector machine [76], naive Bayesian [72] and random forests [11] in a straightforward manner.

Model Learning and Stacking: To minimize $J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})$ in Eq. 5 with respect to $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c}$, we explore the stochastic gradient descent (SGD) technique, which has been shown to perform fairly well in practice [10]. Furthermore, we explore multiple hidden layers to stack multiple model units, in order to generate even more compact and higher-level semantic features resulting in better data

minimization. To improve readability, we defer all the details about model learning and stacking into the appendix, based on which we summarize our deep learning based data minimization mechanism in Algorithm 3.

5.2.2 Perturbation Mechanism for Usage Mode 1 (Achieving ϵ -LDP)

LDP considers the worst-case adversary which can rigorously protect *against all possible inferences* computed over the data (recall Definition 1). In the absence of a user-specified set of sensitive inferences, or otherwise if the user chooses to operate under the LDP guarantees (corresponding to usage mode 1 in Section 1), we develop our perturbation mechanism through perturbing the features learnt from the deep learning based data minimization step in Section 5.2.1. Formally, to achieve ϵ -LDP, DEEProtect under usage mode 1 inserts an Laplacian noise with parameter $\lambda = \sqrt{\dim(\mathbf{f}_t) \dim(\mathbf{x}_t)} \Delta Q / \epsilon$ to each previously learned feature, where $\dim(\mathbf{x}_t), \dim(\mathbf{f}_t)$ represent the dimension of the segmented sensor data \mathbf{x}_t and the features \mathbf{f}_t extracted from \mathbf{x}_t , respectively.³ DEEProtect mechanism under usage mode 1 is summarized in Algorithm 1, which satisfies rigorous ϵ -LDP as will be discussed in Theorem 3.

Our mechanism is different from the baseline approach in Section 5.1 because we add Laplacian noise after the application of the deep learning based data minimization step, while the baseline approach directly adds Laplacian noise to the raw sensor data without the deep learning mechanism. After perturbing these features, we reconstruct the perturbed sensor data according to the decoder function $D_{ec}(\cdot)$ in autoencoder (recall Eq. 3). We will show that our mechanism significantly outperforms the baseline approach (in Sections 6, 8). Note that our privacy objective is also different from that in [2, 75] since they aim to protect each user's training data in the deep learning training stage under the multi-user settings while in contrast we aim to protect the privacy of mobile sensor data stream in single-user settings.

5.2.3 Perturbation Mechanism for Usage Mode 2 (Achieving ϵ -RLDP)

If the user has specified a subset of sensitive inferences (corresponding to usage mode 2 in Section 1), we develop our perturbation mechanism (summarized in Algorithm 2) by first computing the relaxed sensitivity ΔQ_{relax} , and then inserting an Laplacian noise with parameter $\lambda_{\text{relax}} = \sqrt{\dim(\mathbf{f}_t) \dim(\mathbf{x}_t)} \Delta Q_{\text{relax}} / \epsilon$ to each previously learned feature, to satisfy ϵ -RLDP. The detailed process to compute the relaxed sensitivity is as follows.

²Note that such labels are only needed for the training process and are not required when a normal user utilizes our DEEProtect system.

³Although the noise parameter is higher than the baseline approach with a factor of $\sqrt{\dim(\mathbf{f}_t)}$, the number of features, i.e., $\dim(\mathbf{f}_t)$, needed to be perturbed is much smaller than the number of raw sampling points, i.e., $\dim(\mathbf{x}_t)$, in the baseline approach.

Algorithm 1 DEEProtect under Usage Mode 1.

Input: Original Sensor Data $\{\mathbf{x}_t\}_{t=1}^{T/N_w}$; Useful and Sensitive Inference Sets $U(\cdot), S(\cdot)$; Privacy Budget ϵ ;
Output: Perturbed Sensor Data $\{\mathbf{x}'_t\}_{t=1}^{T/N_w}$;
For each $t = 1, 2, \dots, T/N_w$
1. extract features \mathbf{f}_t from \mathbf{x}_t by data minimization mechanism in Algorithm 3;
2. obtain perturbed features \mathbf{f}'_t by inserting noise of $Lap(\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q/\epsilon})$;
3. reconstruct perturbed sensor data $\mathbf{x}'_t = Dec(\mathbf{f}'_t)$;

Computing Relaxed Sensitivity: Similar to *sensitivity* in DP [24], our *relaxed sensitivity* can be computed as

$$\Delta Q_{\text{relax}} = \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}} \|Q(\mathbf{x}_{t1}) - Q(\mathbf{x}_{t2})\|_1 / \dim(\mathbf{x}_t) \quad (6)$$

where $\mathbf{x}_{t1}, \mathbf{x}_{t2}$ are relaxed neighboring databases in Definition 2, and the denominator is set to make *relaxed sensitivity* comparable with traditional *sensitivity* computed over neighboring databases that differ in only one tuple.

The sensitive inferences can be mathematically transformed into a classification problem, which can be addressed by machine learning techniques. The ridge regression classifier in Eq. 4 is a linear function computed over the segmented sensor data \mathbf{x}_t , i.e., $S(\mathbf{x}_t) = \mathbf{x}_t^T \cdot \mathbf{c}^{*S}$. We explain the computation of relaxed sensitivity using this formulation, though our analysis can be generally applied to non-linear situations using kernel-based techniques [20]. We apply the *Gram-Schmidt orthogonalization* technique [9] to a matrix $[\mathbf{c}^{*S}; \mathbf{I}]$ (where \mathbf{I} is an *identity* matrix and thus $[\mathbf{c}^{*S}; \mathbf{I}]$ is full-rank), in order to obtain orthogonal vectors of \mathbf{c}^{*S} as $\mathbf{c}_{\perp 1}^{*S}, \mathbf{c}_{\perp 2}^{*S}, \dots$. Based on that, we form an orthogonal matrix $\mathbf{S} = [\mathbf{c}^{*S}; \mathbf{c}_{\perp 1}^{*S}; \mathbf{c}_{\perp 2}^{*S}; \dots]$. Any function $K(\cdot)$ that is orthogonal of the sensitive inference function $S(\cdot)$ (recall Definition 2) can be represented by a linear combination of $\mathbf{c}_{\perp 1}^{*S}, \mathbf{c}_{\perp 2}^{*S}, \dots$.

For the two relaxed neighboring databases $\mathbf{x}_{t1}, \mathbf{x}_{t2}$, we have $\mathbf{x}_{t1}^T \cdot \mathbf{c}_{\perp 1}^{*S} = \mathbf{x}_{t2}^T \cdot \mathbf{c}_{\perp 1}^{*S}, \mathbf{x}_{t1}^T \cdot \mathbf{c}_{\perp 2}^{*S} = \mathbf{x}_{t2}^T \cdot \mathbf{c}_{\perp 2}^{*S}, \dots$, according to Definition 2. Therefore, we know that $\mathbf{S} \cdot (\mathbf{x}_{t1} - \mathbf{x}_{t2}) = \mathbf{S} \cdot \mathbf{x}_{t1} - \mathbf{S} \cdot \mathbf{x}_{t2} = [\mathbf{x}_{t1}^T \cdot \mathbf{c}^{*S} - \mathbf{x}_{t2}^T \cdot \mathbf{c}^{*S}, \mathbf{x}_{t1}^T \cdot \mathbf{c}_{\perp 1}^{*S} - \mathbf{x}_{t2}^T \cdot \mathbf{c}_{\perp 1}^{*S}, \mathbf{x}_{t1}^T \cdot \mathbf{c}_{\perp 2}^{*S} - \mathbf{x}_{t2}^T \cdot \mathbf{c}_{\perp 2}^{*S}, \dots]^T = [\gamma, 0, 0, \dots]^T$, i.e., $\mathbf{x}_{t1} - \mathbf{x}_{t2} = \mathbf{S}^{-1} \cdot [\gamma, 0, 0, \dots]^T$, and the value of γ is restricted by the range of \mathbf{x}_t . Since we consider the *identity* query (as we publish the sanitized sensor data to mobile apps), we obtain the constraint of γ as $\|\mathbf{S}^{-1} \cdot [\gamma, 0, 0, \dots]^T\|_1 = \|\mathbf{x}_{t1} - \mathbf{x}_{t2}\|_1 \leq \dim(\mathbf{x}_t)\Delta Q$. Therefore, we can compute the relaxed sensitivity as follows.

$$\Delta Q_{\text{relax}} = \max_{\gamma: \|\mathbf{S}^{-1} \cdot [\gamma, 0, 0, \dots]^T\|_1 \leq \dim(\mathbf{x}_t)\Delta Q} \frac{\|\mathbf{S}^{-1} \cdot [\gamma, 0, 0, \dots]^T\|_1}{\dim(\mathbf{x}_t)} \quad (7)$$

It is worth noting that $\Delta Q_{\text{relax}} \leq \Delta Q$ due to the constraint of relaxed neighboring databases for achieving the same

Algorithm 2 DEEProtect under Usage Mode 2.

Input: Original Sensor Data $\{\mathbf{x}_t\}_{t=1}^{T/N_w}$; Useful and Sensitive Inference Sets $U(\cdot), S(\cdot)$; Privacy Budget ϵ ;
Output: Perturbed Sensor Data $\{\mathbf{x}'_t\}_{t=1}^{T/N_w}$;
For each $t = 1, 2, \dots, T/N_w$
1. extract features \mathbf{f}_t from \mathbf{x}_t by data minimization mechanism in Algorithm 3;
2. compute relaxed sensitivity ΔQ_{relax} in Eq. 6;
3. obtain perturbed features \mathbf{f}'_t by inserting noise of $Lap(\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q_{\text{relax}}/\epsilon})$;
4. reconstruct perturbed sensor data $\mathbf{x}'_t = Dec(\mathbf{f}'_t)$;

orthogonal inference values (recall Definition 2). Therefore, comparing to usage mode 1, DEEProtect under usage mode 2 would add less noise to the features extracted in the data minimization step thus achieving better utility.

6 Theoretical Analysis

6.1 Theoretical Privacy Analysis

Our perturbation mechanisms summarized in Algorithms 1, 2 satisfy ϵ -LDP, ϵ -RLDP respectively, according to the following Theorems. To improve readability, we defer the corresponding proofs to the appendix.

Theorem 3 *DEEProtect under usage mode 1 (corresponding to Algorithm 1) satisfies ϵ -LDP.*

Theorem 4 *DEEProtect under usage mode 2 (corresponding to Algorithm 2) satisfies ϵ -RLDP.*

6.2 Theoretical Utility Analysis

For a perturbation algorithm A , let us denote $Error(A) = \mathbb{E}_A[\|A(D) - D\|_1]$ as the expected error in the release of data D , where $\mathbb{E}_A[\cdot]$ is the expectation taken over the randomness of A . We quantify the utility advantage of DEEProtect over the baseline approach in the following Theorems 5, 6 (corresponding to the two usage modes), and the detailed proofs are deferred to the appendix.

Theorem 5 *For DEEProtect under usage mode 1 (corresponding to Algorithm 1), the expected error $Error(A)$ is lower than that of the baseline approach in Section 5.1 by a factor of $\frac{\dim(\mathbf{x}_t)}{\dim(\mathbf{f}_t)}$, where \mathbf{f}_t is the feature set extracted from the segmented sensor data \mathbf{x}_t by using our deep learning based data minimization approach.*

Theorem 6 *For DEEProtect under usage mode 2 (corresponding to Algorithm 2), the expected error $Error(A)$ is lower than that of the baseline approach in Section 5.1 by a factor of $\frac{\dim(\mathbf{x}_t)}{\dim(\mathbf{f}_t)} \cdot \frac{\Delta Q}{\Delta Q_{\text{relax}}}$, where $\Delta Q, \Delta Q_{\text{relax}}$ are the sensitivity and the relaxed sensitivity corresponding to the query function $Q(\cdot)$, respectively.*

Therefore, we can see that the utility advantage of our deep learning based data minimization step in Section 5.2.1 is $\frac{\dim(\mathbf{x}_t)}{\dim(\mathbf{f}_t)}$ and of our relaxed variant of local differential privacy in Section 5.2.3 is $\frac{\Delta Q}{\Delta Q_{\text{relax}}}$.

7 Experimental Setup

In this section, we describe our methodology for collecting sensor data from mobile devices, and the experimental setup (including system parameters) for evaluation.

7.1 Sensor Data Collection

In our experiments, we collected data using a Google Nexus 5 (with 2.3GHz, Krait 400 processor, 16GB internal storage and 2GB RAM on Android 4.4) and a Moto360 smartwatch (with OMAP 3 processor, 4GB internal storage, 512MB RAM on Android Wear OS). On the smartphone, data was captured from the *accelerometer*, *gyroscope*, *orientation*, *ambient light*, *proximity*, and *microphone* sensors. On the smartwatch, the *accelerometer* and *gyroscope* sensors were recorded (recall Figure 2). The sampling rate was fixed at 10 Hz. 20 graduate students (15 males and 5 females) in our university were invited to take our smartphone and smartwatch for two weeks and use them in the same way that they would use their personal devices in their daily lives⁴.

To obtain the ground-truth information for performance evaluation, we ask the users to record labels for both the *useful* and *sensitive inferences*. The labelled training data is grouped under two different categories: mode-detection data and identity-recognition data. Users perform tasks such as walking, enunciating digits or specific alphabets, and the corresponding data segments are then labelled as per the tasks. The mode-detection data, correspond to labelled user activities (e.g., accelerometer data segments are marked with labels such as walking), and speech-to-text translation labels (where audio segments are labelled with the corresponding spoken digit or alphabet). The identity-recognition data is used for authentication and speaker identity recognition experiments. The labelled data is generated by associating the identity of a user as label to a mobile device, on first use.

7.2 System Parameter Setup

We provide provable privacy guarantees for temporal mobile sensor data (discussed above), which is segmented according to the parameter of time window size $N_w = 10$. For the deep learning based data minimization step (in Section 5.2.1), we use 10-fold cross validation to generate the training data and testing data, where 9/10 of

our collected data is used as training data and the remaining 1/10 is used as testing data. We repeated this process for 1000 iterations and reported the averaged results. In our experiments, we used stacked autoencoders ($\alpha = 0.1$ in Algorithm 3) with two hidden layers⁵ comprising of 15 and 7 units respectively. We will show that an autoencoder with only two-hidden layers was able to extract better features than the state-of-the-art techniques. The reduced number of layers (and units) in the autoencoder allowed us to train the model using small amount (2 weeks) of labelled data from the user (note that we are protecting the sensitive inferences for a single user). We implemented all the three case studies (tradeoff between authentication and activity recognition, tradeoff between transportation detection and text recognition, tradeoff between speech translation and speaker identification) discussed in Section 1 on our real-world dataset using the system parameters discussed above.

8 Evaluation

In this section, we experimentally demonstrate the effectiveness of DEEProtect using multiple real-world datasets and applications. We first show the advantage of our deep learning based data minimization step (in Section 5.2.1) over existing feature extraction approaches with up to 2x improvement. Next, we show the advantage of our end-to-end approaches combining both feature extraction and perturbation (in Sections 5.2.2, 5.2.3) over the baseline approach with up to 8x improvement.

8.1 Evaluation for Deep Learning Based Data Minimization Mechanism

We experimentally evaluate our deep learning based data minimization mechanism in Section 5.2.1, using the dataset collected for Case Study 1 (recall Section 1), where the useful inference is the *behavior-based authentication*. To show the advantage of our method, we further compare it with the state-of-the-art feature extraction approaches. The discrete *Fourier* transform (DFT) and discrete cosine transform (DCT) are two basic transformation techniques in the signal processing community, and the Haar basis forms the fundamental wavelet transformation for time-frequency signal analysis. The principal component analysis (PCA) technique [78] can also be utilized to reveal hidden structure in the mobile sensor data. Furthermore, we also consider blind compressive sensing (BCS) as a typical dictionary learning method for comparison [32].

Higher Accuracy: Figure 6 shows the utility-preserving performance under different feature extraction methods. Note that x -axis is in log scale and y -axis is the accuracy

⁴Note that our DEEProtect is not restricted to the usage of these devices and can be generally applied to many real world scenarios.

⁵The two hidden layers are most commonly used in existing deep learning research.

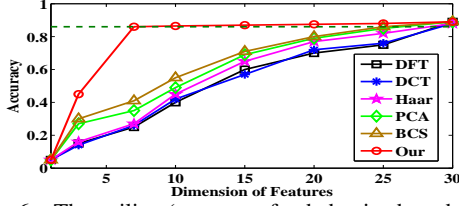


Figure 6. The utility (accuracy for behavior-based authentication in Case Study 1) performance after data minimization shows the advantage of our method over previous approaches.

for behavior-based authentication in Case Study 1 representing the ratio of correctly authenticated users. Note that we use three-dimensional *accelerometer* measurement for behavior-based authentication and set the window size as $N_w = 10$ (recall Section 7.2), therefore the dimension of each segmented sensor data is 3×10 .

From Figure 6, we can see that 1) more features would be beneficial for improving the utility performance since the combination of multiple features would be more accurate and expressive to represent the input data; 2) our method achieves higher accuracy than the state-of-the-art approaches with up to 2x improvement. 7 features are enough for our method to provide good utility performance with 86.13% accuracy, while the accuracy by using all the 30 features is 88.39%. In comparison, the maximum accuracy for the baseline approaches is only 42.01% by using 7 features; 3) our data minimization mechanism significantly benefits from the automatic learning process which explicitly incorporates the useful inference information into deep learning models.

Based on our analysis above, we constrain the dimension of features extracted by our deep learning based data minimization method to 7, whose corresponding accuracy is higher than 97% ($86.13\%/88.39\%$) of the accuracy achieved by using all the features. Therefore, we represent each of the 30-dimensional input sensor data \mathbf{x}_t with a feature set \mathbf{f}_t consisting of only 7 features.

Higher Informativeness: We further leverage *informativeness* in [31] as an important metric to evaluate the information-theoretic relationship between each individual feature and the useful inference information. The *Informativeness* $\text{Info}(f_i)$ of the i -th feature f_i , measures the relative mutual information between the feature and the label of the utility function y^U , and is computed as $\text{Info}(f_i) = \frac{I(f_i; y^U)}{H(y^U)} = 1 - \frac{H(y^U | f_i)}{H(y^U)}$ ⁶. Under the setting of Case Study 1, $y^U = 1, -1$ represents the legitimate user and the adversary, respectively. $H(y^U)$ is the entropy of the variable y^U and $I(f_i; y^U)$ is the mutual information between the random variables of f_i and y^U . For each feature f_i , this measure of *informativeness* takes a value between 0 and 1, where 0 means that the feature contains

⁶ f_i is the random variable taking the i -th value in each \mathbf{f}_t .

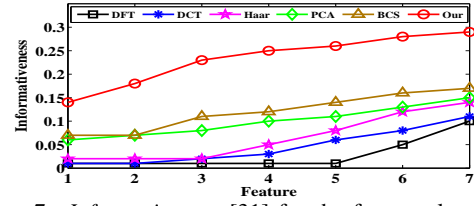


Figure 7. *Informativeness* [31] for the features learned from data minimization. Our method produces better features with higher *informativeness* than the state-of-the-art approaches.

no information about the utility label y^U and 1 means that the feature can completely determine the utility label y^U .

Recall that we set the dimension of features extracted by our method to 7 (see Figure 6). We further evaluate their corresponding *informativeness* in Figure 7. We can observe that the features learned by our method have much higher *informativeness* with up to 2x improvement over previous works. Therefore, our proposed method captures more expressive, higher-quality features than the existing state-of-the-art approaches.

8.2 Evaluation for DEEProtect End-to-end Perturbation Mechanisms

To demonstrate the effectiveness of our end-to-end perturbation mechanisms combining both feature extraction and perturbation, we again consider the *behavior-based authentication* as useful inference and the *activity mode detection* as sensitive inference (recall Case Study 1 in Section 1). We first compare the baseline method with our mechanism under usage mode 1 (corresponding to Algorithm 1), since they achieve the same level of privacy guarantees for preventing all possible sensitive inferences. Then, we compare the baseline method with our mechanism under usage mode 2 (corresponding to Algorithm 2), to verify the effectiveness of our RLDP for protecting a specific subset of sensitive inferences.

Utility Advantage under Both Usage Modes: Figure 8 shows the utility performance computed over the obfuscated sensor data generated by DEEProtect. We can make the following important observations using Figure 8(a): 1) DEEProtect achieves considerable advantage over the baseline approach with up to 8x improvement in utility. This validates the effectiveness of DEEProtect that not only provides rigorous privacy guarantees for protecting sensitive inferences, but also retains the utility of the perturbed data; 2) DEEProtect achieves better utility performance under usage mode 2 (which only considers specific sensitive inferences) than that under usage mode 1 (which considers the entire set of sensitive inferences); 3) As expected, at higher values of ϵ , there is an improvement in utility but at the cost of degradation in privacy; 4) Even at moderate value of $\epsilon = 5$ which is a typical privacy budget in LDP (similar val-

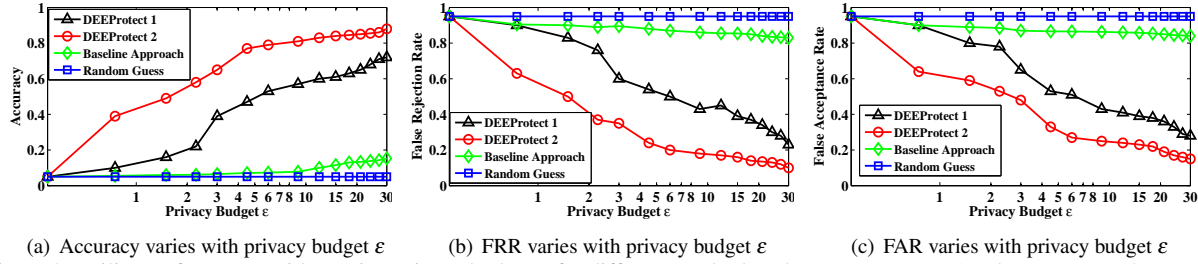


Figure 8. Utility performance with varying privacy budget ϵ for different methods, where *DEEPProtect 1* and *DEEPProtect 2* represent perturbation mechanism under usage mode 1 (Algorithm 1) and usage mode 2 (Algorithm 2), respectively. A larger privacy budget ϵ (less perturbation) results in better utility performance (higher accuracy and lower FRR/FAR). Furthermore, compared to the baseline approach our perturbation mechanism offers a different tradeoff point which allows 8x improvement in utility.

ues can also be found in Google’s Rappor System [30]), the authentication accuracy using DEEPProtect (under usage mode 2) is close to the noise-free level (88.39% in Figure 6). This observation further validates the effectiveness of our mechanisms. Note that the neighboring databases in LDP/RLDP may differ in all their possible tuples (instead of differing in only one tuple as in DP). Thus, a proper privacy budget in LDP/RLDP for balancing utility and privacy is usually higher than that of DP.

The false rejection rate (FRR) in Figure 8(b) is the probability of legitimate users being misclassified as an adversary, and the false acceptance rate (FAR) in Figure 8(c) is the probability of an adversary being misclassified as legitimate users. In Figures 8(b), 8(c), we can observe that our DEEPProtect under both usage modes achieve significantly smaller FRR and FAR compared to the baseline approach (close to the noise-free level).

To investigate the effectiveness of DEEPProtect on real-world mobile applications, we plot the trade-off between the accuracy of making useful inferences versus the accuracy of making sensitive inferences for all the three case studies (corresponding to usage mode 2 and thus Algorithm 2 is applied), as shown in Figure 9.

Utility-Privacy Tradeoffs for Inference-based Access control:

In Figure 9(a), we observe that 1) DEEPProtect achieves good inference performance for behavior-based authentication (useful inference), while significantly deteriorating the activity mode detection (sensitive inference). For privacy budget $\epsilon = 5$ which is a typical privacy budget in LDP (similar values have also been used in Google’s Rappor System [30]), the accuracy for inferring the useful information (authentication) is larger than 80% while the accuracy for inferring sensitive information (activity modes) drops to roughly 5% which is equivalent to random guessing⁷. Therefore, DEEPProtect can achieve practical privacy while only degrading utility performance by 10%. This provides an ef-

fective guide for users to choose a proper value of ϵ for real world applications; 2) higher levels of perturbation would degrade the inference performance for both the useful and sensitive information; 3) DEEPProtect is effective for defending against sensitive inference attacks computed over sensor data. These observations demonstrate that DEEPProtect works well in practice and returns an acceptable utility performance while satisfying provable privacy guarantees.

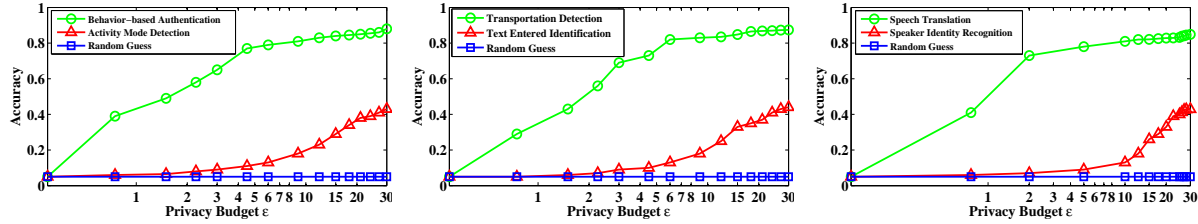
Similarly, for the other two case studies, the accuracy of the sensitive inferences degrades at a much faster rate than that of the useful inferences when more noise is added (corresponding to a smaller privacy budget ϵ), which further validates the effectiveness of DEEPProtect. For Case Study 2, in Figure 9(b), when $\epsilon \geq 5$, DEEPProtect achieves good performance for transportation detection (useful inference), while significantly degrading the identification of entered text (sensitive inference). For Case Study 3, in Figure 9(c), when $\epsilon \geq 3$, DEEPProtect achieves good performance for speech translation (useful inference), while preventing the recognition of speaker identity (sensitive inference).

9 Discussions and Limitations

9.1 Privacy Considerations

Depending on the mode the user chooses to operate in DEEPProtect can provide either the rigorous LDP guarantee that protects all sensitive inferences (usage mode 1) or our RLDP that protects a user-chosen subset of inferences (usage mode 2). While RLDP provides weaker privacy guarantees than LDP, it comes with the advantage of stronger utility properties. When applying LDP/RLDP on mobile sensor data, the neighboring databases may differ in all their possible tuples (i.e., all sensor recordings across all timestamps within the same window), while the neighboring databases in traditional DP only differ in one tuple. Therefore, a proper privacy budget in LDP/RLDP for balancing privacy and utility is usually higher than that of DP [23, 30, 68]. Previous work in [67, 75] proposed to ignore several records in

⁷Since we collected 20 users’ data in our experiments, the accuracy for random guessing is $1/20 = 5\%$.



(a) Utility-Privacy tradeoff for Case Study 1 (b) Utility-Privacy tradeoff for Case Study 2 (c) Utility-Privacy tradeoff for Case Study 3
Figure 9. The trade-off between utility and privacy for the three case studies in Section 1. Compared to the useful inferences (shown as green lines), the degradation in accuracy of the sensitive inference (shown as red lines) is much faster. For $\epsilon = 5$ in Figure 9(a), 9(b) and $\epsilon = 3$ in Figure 9(c), the accuracy for the useful inference is close to the noise free scenario (less than 10% degradation) while the sensitive inference is close to a random guess.

the database to bound sensitivity in DP for providing better privacy-utility tradeoffs. Our approach of using deep learning techniques to extract authorized features is a conceptual improvement over these methods. While DEEPProtect has been presented in the context of privacy-aware data sharing on mobile phones, the techniques developed are not restricted to the specific setting, and can be applied to other scenarios beyond mobile phones. We will make DEEPProtect tool available to public as open source software.

Although our privacy guarantee is limited to segmented sensor data, we do take the temporal dynamics existing in mobile sensor stream into consideration according to a user-specified parameter of window size. Similar to any DP-oriented metrics, our privacy guarantees also compose securely, i.e., retain privacy guarantees even in the presence of multiple independent releases (recall Theorems 1, 2). While there is a non-trivial utility cost incurred by our mechanisms, DEEPProtect (under both usage modes) significantly outperforms the baseline approach. In addition, our utility performance can be further improved by 1) exploiting fine-grained correlation among sensors (e.g., among the three dimensions of accelerometer in Case Study 1) according to [51]; 2) training customized models specific to each individual user that installs DEEPProtect in phone; and 3) training models based on a larger amount of users’ data as in [30]. Our perturbation mechanism can also be easily generalized to consider correlation across time windows, according to the composition properties of our privacy metrics (refer to [23] and Theorems 1,2). An important component of our perturbation mechanism is the computation of the feature sensitivity in Algorithms 1,2. Ways to accurately compute the sensitivity for arbitrary mobile applications, and deal with its possible underestimation would be a challenge we would like to address in the future.

9.2 Deployment Considerations

We are working towards an implementation of the auto-encoder on the Android platform using one of the deep

learning packages on Android [1,41]. The training itself could be done offline and then transferred on the phone to perform data minimization in real time [44]. Our off-line training process requires a small amount of labeled data, for example, in our experiments, we only required 20 users’ data collected for 2 weeks.

Any DP-oriented mechanism that adds noise to the data (or corresponding query) can risk the problem of breaking the integrity constraints of these sensors. In practice, since our framework targets simple sensors such as accelerometer, gyroscope, etc., our decoding process is unlikely to break the integrity constraints for these sensors. Therefore, apps can continue to use the perturbed sensor data generated by DEEPProtect without any change to their existing interface.

10 Conclusions

In this paper, we propose DEEPProtect, a general privacy-preserving framework for inference-based access control of time-series sensor data on mobile devices. DEEPProtect not only supports conventional LDP guarantees, but also provides a novel relaxed variant of LDP. We further propose effective perturbation mechanisms consisting of two key steps: 1) we uniquely explore deep learning techniques to realize data minimization and only retain features relevant to the useful (authorized) inferences. This prevents the leakage of any information that is orthogonal to the useful inferences; and 2) to further enhance the privacy of sensitive inferences, we perturb the extracted features, using an effective obfuscation mechanism that ensures both conventional and relaxed LDP guarantees while simultaneously maintaining the utility of the data. Finally, for reasons of compatibility with existing third-party apps, we reconstruct the sensor data, from the noisy features before sharing. Through theoretical analysis and extensive experiments over multiple real-world datasets, we demonstrate that compared to the state-of-the-art research, DEEPProtect significantly improves the accuracy for supporting mobile sensing applications while providing provable privacy guarantees.

11 Acknowledgement

This work has been partly supported by faculty awards from Google, Intel and NSF. Changchang Liu is partly supported by IBM PhD Fellowship. We are very thankful to Brendan McMahan, Keith Bonawitz, Daniel Ramage, Nina Taft from Google, and Richard Chow from Intel for useful feedback and discussions.

References

- [1] Torch for android. <https://github.com/soumith/torch-android>.
- [2] ABADI, M., CHU, A., GOODFELLOW, I., MCMAHAN, H. B., MIRONOV, I., TALWAR, K., AND ZHANG, L. Deep learning with differential privacy. In *ACM CCS* (2016).
- [3] AZIZYAN, M., CONSTANDACHE, I., AND ROY CHOUDHURY, R. Surroundsense: mobile phone localization via ambience fingerprinting. In *Mobicom* (2009).
- [4] BACKES, M., BUGIEL, S., HAMMER, C., SCHRANZ, O., AND VON STYP-REKOWSKY, P. Boxify: Full-fledged app sandboxing for stock android. In *Usenix Security* (2015).
- [5] BAO, L., AND INTILLE, S. S. Activity recognition from user-annotated acceleration data. In *PerCom*. 2004.
- [6] BASCH, J., GUIBAS, L. J., AND HERSHBERGER, J. Data structures for mobile data. *Journal of Algorithms* (1999).
- [7] BERESFORD, A. R., RICE, A., SKEHIN, N., AND SOHAN, R. Mockdroid: trading privacy for application functionality on smartphones. In *HotMobile* (2011).
- [8] BERTSEKAS, D. P., AND TSITSIKLIS, J. N. Neuro-dynamic programming: an overview. In *CDC* (1995).
- [9] BJÖRCK, Å. Solving linear least squares problems by gram-schmidt orthogonalization. *BIT Numerical Mathematics* (1967).
- [10] BOTTOU, L. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nimes* (1991).
- [11] BREIMAN, L. Random forests. *Machine learning* (2001).
- [12] BROUWERS, N., AND WOEHRLE, M. Detecting dwelling in urban environments using gps, wifi, and geolocation measurements. In *PhoneSense* (2011).
- [13] CAI, L., AND CHEN, H. Touchlogger: Inferring keystrokes on touch screen from smartphone motion. In *HotSec* (2011).
- [14] CAVOUKIAN, A., ET AL. Privacy by design: The 7 foundational principles. *Information and Privacy Commissioner of Ontario, Canada* (2009).
- [15] CHAKRABORTY, S., SHEN, C., RAGHAVAN, K. R., SHOUKRY, Y., MILLAR, M., AND SRIVASTAVA, M. ipshield: a framework for enforcing context-aware privacy. In *NSDI* (2014).
- [16] CHANG, K.-H., FISHER, D., AND CANNY, J. Ammon: A speech analysis library for analyzing affect, stress, and mental health on mobile phones. *PhoneSense* (2011).
- [17] CHEN, R., FUNG, B. C., PHILIP, S. Y., AND DESAI, B. C. Correlated network data publication via differential privacy. *The VLDB Journal* (2014).
- [18] CONTI, M., ZACHIA-ZLATEA, I., AND CRISPO, B. Mind how you answer me!: transparently authenticating the user of a smartphone when answering or placing a call. In *AsiaCCS* (2011).
- [19] CORNELIUS, C., KAPADIA, A., KOTZ, D., PEEBLES, D., SHIN, M., AND TRIANOPOULOS, N. Anonymsense: privacy-aware people-centric sensing. In *MobiSys* (2008).
- [20] CRISTIANINI, N., AND SHAW-TAYLOR, J. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.
- [21] DE CRISTOFARO, E., AND DI PIETRO, R. Preserving query privacy in urban sensing systems. In *Distributed Computing and Networking*. 2012.
- [22] DE CRISTOFARO, E., AND SORIENTE, C. Participatory privacy: Enabling privacy in participatory sensing. *Network, IEEE* (2013).
- [23] DUCHI, J. C., JORDAN, M. I., AND WAINWRIGHT, M. J. Local privacy and statistical minimax rates. In *FOCS* (2013).

- [24] DWORK, C. Differential privacy. In *Automata, languages and programming*. Springer, 2006.
- [25] DWORK, C. Differential privacy: A survey of results. In *Theory and Applications of Models of Computation*. 2008.
- [26] DWORK, C. A firm foundation for private data analysis. *Communications of the ACM* (2011).
- [27] DWORK, C., MCSHERRY, F., NISSIM, K., AND SMITH, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography*. Springer, 2006.
- [28] DWORK, C., ROTH, A., ET AL. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science* (2014).
- [29] DWORK, C., AND SMITH, A. Differential privacy for statistics: What we know and what we want to learn. *Journal of Privacy and Confidentiality* (2010).
- [30] ERLINGSSON, Ú., PIHUR, V., AND KOROLOVA, A. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS* (2014).
- [31] FRANK, M., BIEDERT, R., MA, E.-D., MARTINOVIC, I., AND SONG, D. Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication. *IEEE TIFS* (2013).
- [32] GLEICHMAN, S., AND ELDAR, Y. C. Blind compressed sensing. *IEEE TIT* (2011).
- [33] GÖTZ, M., NATH, S., AND GEHRKE, J. Maskit: Privately releasing user context streams for personalized mobile applications. In *SIGMOD* (2012).
- [34] HAN, J., OWUSU, E., NGUYEN, L. T., PERRIG, A., AND ZHANG, J. Accomplice: Location inference using accelerometers on smartphones. In *COMSNETS* (2012).
- [35] HE, X., MACHANAVAJJHALA, A., AND DING, B. Blowfish privacy: Tuning privacy-utility trade-offs using policies. In *SIGMOD* (2014).
- [36] HEMMINKI, S., NURMI, P., AND TARKOMA, S. Accelerometer-based transportation mode detection on smartphones. In *SenSys* (2013).
- [37] HINTON, G. E., AND SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. *Science* (2006).
- [38] HOERL, A. E., AND KENNARD, R. W. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics* (1970).
- [39] HORNYACK, P., HAN, S., JUNG, J., SCHECHTER, S., AND WETHERALL, D. These aren't the droids you're looking for: retrofitting android to protect data from imperious applications. In *CCS* (2011).
- [40] HOUSE, W. Consumer data privacy in a networked world: A framework for protecting privacy and promoting innovation in the global digital economy. *White House, Washington, DC* (2012).
- [41] JIA, Y., SHELHAMER, E., DONAHUE, J., KARAYEV, S., LONG, J., GIRSHICK, R., GUADARRAMA, S., AND DARRELL, T. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093* (2014).
- [42] KIFER, D., AND MACHANAVAJJHALA, A. Pufferfish: A framework for mathematical privacy definitions. *ACM TODS* (2014).
- [43] KIM, D. H., KIM, Y., ESTRIN, D., AND SRIVASTAVA, M. B. Sensloc: sensing everyday places and paths using less energy. In *SenSys* (2010).
- [44] LANE, N. D., AND GEORGIEV, P. Can deep learning revolutionize mobile sensing? In *HotMobile* (2015).
- [45] LEE, H., EKANADHAM, C., AND NG, A. Y. Sparse deep belief net model for visual area v2. In *NIPS* (2008).
- [46] LEE, W.-H., AND LEE, R. B. Multi-sensor authentication to improve smartphone security. In *ICISSP* (2015).
- [47] LEI, X., SENIOR, A., GRUENSTEIN, A., AND SORESENSEN, J. Accurate and compact large vocabulary speech recognition on mobile devices. In *INTERSPEECH* (2013), pp. 662–665.
- [48] LI, Q., AND CAO, G. Efficient privacy-preserving stream aggregation in mobile sensing with low aggregation error. In *PETS* (2013).
- [49] LI, S., DA XU, L., AND WANG, X. Compressed sensing signal and data acquisition in wireless sensor networks and internet of things. *IEEE Transactions on Industrial Informatics* (2013).
- [50] LIU, B., JIANG, Y., SHA, F., AND GOVINDAN, R. Cloud-enabled privacy-preserving collaborative learning for mobile sensing. In *SenSys* (2012).
- [51] LIU, C., CHAKRABORTY, S., AND MITTAL, P. Dependence makes you vulnerable: Differential privacy under dependent tuples. In *NDSS* (2016).
- [52] LIU, X., ZHOU, Z., DIAO, W., LI, Z., AND ZHANG, K. When good becomes evil: Keystroke inference with smartwatch. In *CCS* (2015).
- [53] LU, H., FRAUENDORFER, D., RABBI, M., MAST, M. S., CHITTARANJAN, G. T., CAMPBELL, A. T., GATICA-PEREZ, D., AND CHOUDHURY, T. Stresssense: Detecting stress in unconstrained acoustic environments using smartphones. In *UbiComp* (2012).
- [54] LUO, C., WU, F., SUN, J., AND CHEN, C. W. Compressive data gathering for large-scale wireless sensor networks. In *MobiCom* (2009).
- [55] LUXTON, D. D., MCCANN, R. A., BUSH, N. E., MISHKIND, M. C., AND REGER, G. M. mhealth for mental health: Integrating smartphone technology in behavioral healthcare. *Professional Psychology: Research and Practice* (2011).
- [56] MARE, S., MARKHAM, A. M., CORNELIUS, C., PETERSON, R., AND KOTZ, D. Zebra: Zero-effort bilateral recurring authentication. In *IEEE S&P* (2014).
- [57] MARQUARDT, P., VERMA, A., CARTER, H., AND TRAYNOR, P. (sp) iphone: decoding vibrations from nearby keyboards using mobile phone accelerometers. In *CCS* (2011).
- [58] MCSHERRY, F. D. Privacy integrated queries: an extensible platform for privacy-preserving data analysis. In *SIGMOD* (2009).
- [59] MICHALEVSKY, Y., BONEH, D., AND NAKIBLY, G. Gyrophone: Recognizing speech from gyroscope signals. In *USENIX Security* (2014).
- [60] MILUZZO, E., VARSHAVSKY, A., BALAKRISHNAN, S., AND CHOUDHURY, R. R. Tapprints: your finger taps have fingerprints. In *MobiSys* (2012).
- [61] MOHAN, P., PADMANABHAN, V. N., AND RAMJEE, R. Nericell: rich monitoring of road and traffic conditions using mobile smartphones. In *SenSys* (2008).
- [62] NAM, W., DOLLÁR, P., AND HAN, J. H. Local decorrelation for improved pedestrian detection. In *NIPS* (2014).
- [63] NIRJON, S., DICKERSON, R. F., ASARE, P., LI, Q., HONG, D., STANKOVIC, J. A., HU, P., SHEN, G., AND JIANG, X. Auditeur: A mobile-cloud service platform for acoustic event detection on smartphones. In *MobiSys* (2013).
- [64] OWUSU, E., HAN, J., DAS, S., PERRIG, A., AND ZHANG, J. Accessory: password inference using accelerometers on smartphones. In *HotMobile* (2012).
- [65] PARK, J.-G., PATEL, A., CURTIS, D., TELLER, S., AND LEDLIE, J. Online pose classification and walking speed estimation using handheld devices. In *UbiComp* (2012).

- [66] PFITZMANN, A., AND HANSEN, M. A terminology for talking about privacy by data minimization: Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. In http://dud.inf.tu-dresden.de/literatur/Anon_Terminology_v0.33.pdf (2010).
- [67] PROSERPIO, D., GOLDBERG, S., AND MCSHERRY, F. Calibrating data to sensitivity in private data analysis: a platform for differentially-private analysis of weighted datasets. In *VLDB* (2014).
- [68] QIN, Z., YANG, Y., YU, T., KHALIL, I., XIAO, X., AND REN, K. Heavy hitter estimation over set-valued data with local differential privacy. In *CCS* (2016).
- [69] RACHURI, K. K., MUSOLESI, M., MASCOLO, C., RENTFROW, P. J., LONGWORTH, C., AND AUCINAS, A. Emotionsense: a mobile phones based adaptive platform for experimental social psychology research. In *UbiComp* (2010).
- [70] RASTOGI, V., AND NATH, S. Differentially private aggregation of distributed time-series with transformation and encryption. In *SIGMOD* (2010).
- [71] REDDY, S., MUN, M., BURKE, J., ESTRIN, D., HANSEN, M., AND SRIVASTAVA, M. Using mobile phones to determine transportation modes. *ACM TOSN* (2010).
- [72] RISH, I. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence* (2001).
- [73] RIVA, O., QIN, C., STRAUSS, K., AND LYMBERPOULOS, D. Progressive authentication: Deciding when to authenticate on mobile phones. In *USENIX Security* (2012).
- [74] SHEBARO, B., OGUNWUYI, O., MIDI, D., AND BERTINO, E. Identidroid: Android can finally wear its anonymous suit. *ACM TODP* (2014).
- [75] SHOKRI, R., AND SHMATIKOV, V. Privacy-preserving deep learning. In *CCS* (2015).
- [76] SUYKENS, J. A., AND VANDEWALLE, J. Least squares support vector machine classifiers. *Neural processing letters* (1999).
- [77] TEMPLEMAN, R., RAHMAN, Z., CRANDALL, D., AND KAPADIA, A. Placeraider: Virtual theft in physical spaces with smartphones. In *NDSS* (2013).
- [78] TIPPING, M. E., AND BISHOP, C. M. Probabilistic principal component analysis. *Journal of the Royal Statistical Society* (1999).
- [79] TSOCHANTARIDIS, I., HOFMANN, T., JOACHIMS, T., AND ALTUN, Y. Support vector machine learning for interdependent and structured output spaces. In *ICML* (2004).
- [80] TUNG, Y.-C., AND SHIN, K. G. Echotag: Accurate infrastructure-free indoor location tagging with smartphones. In *MobiCom* (2015).
- [81] VAN DER MAATEN, L., POSTMA, E., AND VAN DEN HERIK, J. Dimensionality reduction: a comparative. *J Mach Learn Res* 10 (2009), 66–71.
- [82] VINCENT, P., LAROCHELLE, H., BENGIO, Y., AND MANZAGOL, P.-A. Extracting and composing robust features with denoising autoencoders. In *ICML* (2008).
- [83] WANG, R., ENCK, W., REEVES, D., ZHANG, X., NING, P., XU, D., ZHOU, W., AND AZAB, A. M. Easeandroid: Automatic policy analysis and refinement for security enhanced android via large-scale semi-supervised learning. In *USENIX Security* (2015).
- [84] WARNER, S. L. Randomized response: A survey technique for eliminating evasive answer bias. *Journal of the American Statistical Association* (1965).

- [85] XU, Z., BAI, K., AND ZHU, S. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *WiSec* (2012).
- [86] ZHOU, B., CUI, X., HUANG, S., CMEJREK, M., ZHANG, W., XUE, J., CUI, J., XIANG, B., DAGGETT, G., CHAUDHARI, U., ET AL. The ibm speech-to-speech translation system for smartphone: Improvements for resource-constrained tasks. *Computer Speech & Language* (2013).
- [87] ZHU, J., WU, P., WANG, X., AND ZHANG, J. Sensec: Mobile security through passive sensing. In *ICNC* (2013).

12 Appendix

12.1 Composition Properties of RLDP

Sequential Composition Theorem: For D_1 and D_2 , let $D_{1t} = D_1 \cap D_{1t}$ and $D_{2t} = D_2 \cap D_{2t}$. For any sequence r of outcomes $r_t \in \mathbb{R}(A_t)$, the probability of output r from the sequence of $A_t(D_1)$ is $\mathbb{P}(\tilde{A}(D_1) = r) = \prod_t \mathbb{P}(A_t(D_{1t}) = r_t)$. Applying the definition of RLDP for each A_t , we have $\prod_t \mathbb{P}(A_t(D_{1t}) = r_t) \leq \prod_t \mathbb{P}(A_t(D_{2t}) = r_t) \times \prod_t \exp(\frac{\epsilon_t}{\Delta D} \times |D_{1t} - D_{2t}|) \leq \mathbb{P}(\tilde{A}(D_2) = r) \times \exp(\frac{\sum_t \epsilon_t}{\Delta D} \times |D_1 - D_2|) \leq \mathbb{P}(\tilde{A}(D_2) = r) \times \exp(\sum_t \epsilon_t)$.

Parallel Composition Theorem: For D_1 and D_2 , let $D_{1t} = D_1 \cup D_{1t}$ and $D_{2t} = D_2 \cup D_{2t}$. For any sequence r of outcomes $r_t \in \mathbb{R}(A_t)$, the probability of output r from the sequence of $A_t(D_1)$ is $\mathbb{P}(\tilde{A}(D_1) = r) = \prod_t \mathbb{P}(A_t(D_{1t}) = r_t)$. Applying the definition of RLDP for each A_t , we have $\prod_t \mathbb{P}(A_t(D_{1t}) = r_t) \leq \prod_t \Pr(A_t(D_{2t}) = r_t) \times \prod_t \exp(\frac{\epsilon_t}{\Delta D} \times |D_{2t} - D_{1t}|) \leq \mathbb{P}(\tilde{A}(D_2) = r) \times \exp(\frac{\max_t \epsilon_t}{\Delta D} \times |D_1 - D_2|) \leq \mathbb{P}(\tilde{A}(D_2) = r) \times \exp(\max_t \epsilon_t)$.

12.2 Privacy Guarantees for the Baseline Approach

For the baseline approach that adds Laplace noise $\lambda = \dim(\mathbf{x}_t)\Delta Q/\epsilon$ to the sensor data, we have $\max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\mathbb{P}(A(\mathbf{x}_{t1})=o)}{\mathbb{P}(A(\mathbf{x}_{t2})=o)} \leq \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\text{Lap}(o-Q(\mathbf{x}_{t1}))}{\text{Lap}(o-Q(\mathbf{x}_{t2}))} \leq \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\exp(\frac{\epsilon(o-Q(\mathbf{x}_{t1}))}{\dim(\mathbf{x}_t)\Delta Q})}{\exp(\frac{\epsilon(o-Q(\mathbf{x}_{t2}))}{\dim(\mathbf{x}_t)\Delta Q})} \leq \exp(\epsilon)$, since $\max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}} \|Q(\mathbf{x}_{t1}) - Q(\mathbf{x}_{t2})\| = \dim(\mathbf{x}_t)\Delta Q$ (composition properties of DP).

12.3 Theoretical Analysis of DEEProtect Perturbation Mechanisms

Proof for Theorem 3: First, we have $\Delta \mathbf{f}_t \leq \sqrt{\dim(\mathbf{f}_t)\Delta_2 \mathbf{f}_t} \leq \sqrt{\dim(\mathbf{f}_t)\Delta_2 \mathbf{x}_t} \leq \sqrt{\dim(\mathbf{f}_t)\Delta \mathbf{x}_t} \leq \sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q}$. Therefore, by setting $\lambda = \sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q}/\epsilon$, we have $\max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\mathbb{P}(A(\mathbf{x}_{t1})=o)}{\mathbb{P}(A(\mathbf{x}_{t2})=o)} \leq \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\text{Lap}(o-Q(\mathbf{x}_{t1}))}{\text{Lap}(o-Q(\mathbf{x}_{t2}))} \leq$

Algorithm 3 Deep Learning based Data Minimization Mechanism

Input: The sensor data $\{\mathbf{x}_t\}_{t=1}^{T/N_w}$, the learning rate α ;

Output: Weight matrix \mathbf{W} , \mathbf{W}' and bias $\mathbf{b}_e, \mathbf{b}_d$;

1. Initialize $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{c}$ randomly and set the tied weights $\mathbf{W}' = \mathbf{W}^T$ according to [37, 45, 82];
2. **For each iteration** = 1, 2, 3... **do**
3. Set $\Delta\mathbf{W} = 0, \Delta\mathbf{b}_e = 0, \Delta\mathbf{b}_d = 0, \Delta\mathbf{c} = 0$;
4. Compute the partial derivatives:

$$\begin{aligned}\Delta\mathbf{W} &= \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{W}} \\ \Delta\mathbf{b}_e &= \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{b}_e} \\ \Delta\mathbf{b}_d &= \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{b}_d} \\ \Delta\mathbf{c} &= \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{c}}\end{aligned}\quad (9)$$

5. Update $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{c}$ by gradient descent as

$$\begin{aligned}\mathbf{W} &:= \mathbf{W} - \alpha\Delta\mathbf{W}, & \mathbf{b}_e &:= \mathbf{b}_e - \alpha\Delta\mathbf{b}_e \\ \mathbf{b}_d &:= \mathbf{b}_d - \alpha\Delta\mathbf{b}_d, & \mathbf{c} &:= \mathbf{c} - \alpha\Delta\mathbf{c}\end{aligned}\quad (10)$$

6. Orthogonalize the weight matrix as

$$\mathbf{W} := (\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}}\mathbf{W}\quad (11)$$

$$\max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\exp\left(\frac{\varepsilon(o-Q(\mathbf{x}_{t1}))}{\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q}}\right)}{\exp\left(\frac{\varepsilon(o-Q(\mathbf{x}_{t2}))}{\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q}}\right)} \leq \exp(\varepsilon).$$

Proof for Theorem 4: By setting $\lambda = \sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q_{\text{relaxed}}}/\varepsilon$, we have

$$\begin{aligned}\max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\mathbb{P}(A(\mathbf{x}_{t1})=o)}{\mathbb{P}(A(\mathbf{x}_{t2})=o)} &\leq \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\text{Lap}(o-Q(\mathbf{x}_{t1}))}{\text{Lap}(o-Q(\mathbf{x}_{t2}))} \leq \\ \max_{\mathbf{x}_{t1}, \mathbf{x}_{t2}, o} \frac{\exp\left(\frac{\varepsilon(o-Q(\mathbf{x}_{t1}))}{\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q_{\text{relaxed}}}}\right)}{\exp\left(\frac{\varepsilon(o-Q(\mathbf{x}_{t2}))}{\sqrt{\dim(\mathbf{f}_t)\dim(\mathbf{x}_t)\Delta Q_{\text{relaxed}}}}\right)} &\leq \exp(\varepsilon).\end{aligned}$$

Proof for Theorem 5: First, we derive the variance of a randomized algorithm $A(\cdot)$ for our DEEPProtect under usage mode 1 as $\text{Var}(A(\mathbf{x}_t)) = \sum_{i=1}^{\dim(\mathbf{f}_t)} \frac{\text{Var}(\mathbf{f}_i')}{\dim(\mathbf{x}_t)^2} \leq \frac{\dim(\mathbf{f}_t)^2(\Delta\mathbf{f}_i(i))^2/\varepsilon^2}{\dim(\mathbf{x}_t)^2} \leq \frac{\dim(\mathbf{f}_t)^2\dim(\mathbf{f}_t)^2(\Delta Q)^2/\varepsilon^2}{\dim(\mathbf{x}_t)^2} \leq \frac{\dim(\mathbf{f}_t)^2(\Delta Q)^2}{\dim(\mathbf{x}_t)^2\varepsilon^2}$. Then, we compute the expected error as $\mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|] \leq \mathbb{E}[\|A(\mathbf{x}_t) - \mathbb{E}[A(\mathbf{x}_t)]\|_1] + \mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|_1] = \text{Re_error}(A(\mathbf{x}_t)) + \sqrt{\mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|_2^2]} = \text{Re_error}(A(\mathbf{x}_t)) + \sqrt{\text{Var}(A(\mathbf{x}_t))}$. Note that our deep learning based data minimization mechanism would result in a negligible reconstruction error, which makes it fairly applicable in many practical scenarios. It is likely that the reconstruction error $\text{Re_error}(A(\mathbf{x}_t))$

is much lower than the perturbation error caused by adding noise. Therefore, we approximate the utility performance of DEEPProtect as $\text{Error}(A) \approx \frac{\dim(\mathbf{f}_t)\Delta Q}{\dim(\mathbf{x}_t)\varepsilon}$ (similar results can also be found in [70]). Similarly, we evaluate the expected error for the baseline approach as $\mathbb{E}[\|LPM(\mathbf{x}_t) - \mathbf{x}_t\|_1] = \mathbb{E}[\|(Lap(\Delta Q/\varepsilon))\|_1] = \Delta Q/\varepsilon$. Comparing the utility performance for both methods, we know that DEEPProtect under usage mode 1 reduces the expected error of the baseline approach with a factor of $\dim(\mathbf{x}_t)/\dim(\mathbf{f}_t)$.

Proof for Theorem 6: First, we derive the variance of a randomized algorithm $A(\cdot)$ for our DEEPProtect under usage mode 2 as $\mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|] \leq \mathbb{E}[\|A(\mathbf{x}_t) - \mathbb{E}[A(\mathbf{x}_t)]\|_1] + \mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|_1] = \text{Re_error}(A(\mathbf{x}_t)) + \sqrt{\mathbb{E}[\|A(\mathbf{x}_t) - \mathbf{x}_t\|_2^2]} = \text{Re_error}(A(\mathbf{x}_t)) + \sqrt{\text{Var}(A(\mathbf{x}_t))}$.

Similarly, we have $\text{Error}(A) \approx \frac{\dim(\mathbf{f}_t)\Delta Q_{\text{relaxed}}}{\dim(\mathbf{x}_t)\varepsilon}$. Comparing the utility performance for both methods, we know that DEEPProtect under usage mode 2 reduces the expected error of the baseline approach with a factor of $\dim(\mathbf{x}_t)\Delta Q/\dim(\mathbf{f}_t)\Delta Q_{\text{relaxed}}$.

12.4 Model Learning and Stacking in Deep Learning Based Data Minimization

Existing Autoencoder Models: The mathematical formulations of existing autoencoder models [37, 45] (expanding Eq. 3) is

$$\begin{aligned}\min J_0(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}') &= \min \sum_{t=1}^{T/N_w} L(\mathbf{x}_t, D_{ec}(E_{nc}(\mathbf{x}_t))) \\ &+ \lambda \sum_{t,i} \mathbf{W}_{t,i}^2 + \delta \sum_{i=1}^{d_f} KL(\rho \|\hat{\rho}_i)\end{aligned}\quad (8)$$

where the encoder function $E_{nc}(\cdot)$ maps the input data $\mathbf{x}_t \in \mathbb{R}^{d_{\mathbf{x}} \times 1}$ to the hidden units (features) $\mathbf{f}_t \in \mathbb{R}^{d_{\mathbf{f}} \times 1}$ according to $\mathbf{f}_t = E_{nc}(\mathbf{x}_t) = g_{enc}(\mathbf{W}\mathbf{x}_t + \mathbf{b}_e)$, and the decoder function $D_{ec}(\cdot)$ maps the outputs of the hidden units (features) back to the original input space according to $\tilde{\mathbf{x}}_t = D_{ec}(\mathbf{f}_t) = g_{dec}(\mathbf{W}'\mathbf{f}_t + \mathbf{b}_d)$. Similar to most existing deep learning methods [37, 45, 82], we choose the sigmoid function for both the encoding activation function $g(\cdot)$ and the decoding activation function $g_{dec}(\cdot)$, and we focus on the tied weights case in which $\mathbf{W}' = \mathbf{W}^T$ (where \mathbf{W}^T is the transpose of \mathbf{W}). $L(\mathbf{u}, \mathbf{v})$ is a loss function, typically the square loss $L(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|^2$. The second term in Eq. 8 is a weight decay term that helps prevent overfitting [37]. The third term in Eq. 8 is a sparsity constraint $\sum_{i=1}^{d_f} KL(\rho \|\hat{\rho}_i)$ with a pre-determined sparsity parameter ρ , where $\hat{\rho}_i = \frac{1}{n} \sum_{t=1}^{T/N_w} f_i(\mathbf{x}_t)$ is the average activation of hidden unit i and $KL(\rho \|\hat{\rho}_i) = \rho \log \frac{\rho}{\hat{\rho}_i} + (1 - \rho) \log \frac{1-\rho}{1-\hat{\rho}_i}$ is the KL divergence between two Bernoulli random variables with mean ρ and $\hat{\rho}_i$ respectively.

Model Learning: The objective for our data minimization mechanism is to minimize $J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})$ in

Table 1. Summary of Existing Inferences Computed over Mobile Sensor Data.

	Accelerometer	Orientation	Gyroscope	Magnetic Field	Microphone	Camera	GPS	Touch Screen
Activity Mode Detection <i>attribute: walking, still, etc.</i>	[5, 36, 71]							
Behavior-based Authentication <i>attribute: authorized user or not</i>	[18, 46, 56, 73, 87]	[18, 46]	[56, 87]	[46, 87]	[73]	[73]		[31, 73]
Text Entered <i>attribute: alphabets, digits</i>	[57, 60, 64, 85]	[13, 85]	[60]					
Speaker Identity Recognition <i>attribute: user's identity</i>	[50]	[50]	[50]	[50]	[50]	[50, 63]		
Speech-to-text Translation <i>attribute: speech to text</i>			[59]			[47, 86]		
Location <i>attribute: home, work, public</i>	[34, 43]			[63]			[12, 43]	
Device Placement <i>attribute: hand, ear, pocket, bag</i>	[65]							
Onscreen Taps <i>attribute: location of apps on screen</i>	[60]		[60]					
Stress <i>attribute: stressful or not</i>					[16, 53]			
Emotion <i>attribute: happy, sad, fear, anger</i>					[16, 69]			
Environment Monitor <i>attribute: place virtualization</i>	[61, 77]	[77]			[3, 61, 77]	[3, 61, 77]		

Eq. 5 with respect to $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c}$. We explore the stochastic gradient descent (SGD) technique to solve this optimization problem, which has been shown to perform fairly well in practice [10]. Note that we use SGD to only solve the convex optimization problem in Eq. 5 (without considering the non-convex constraint of $\mathbf{W}\mathbf{W}^T = \mathbf{I}$). To train our neural network, we first initialize each parameter $\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c}$ to a small random value near zero, and then apply SGD for iterative optimization. Before computing the exact gradient for the objective function in Eq. 5 with respect to each variable, we first perform the feedforward pass and evaluate $\mathbf{z}_t^e = \mathbf{W}\mathbf{d}_t + \mathbf{b}_e, \mathbf{z}_t^d = \mathbf{W}^T s_e(\mathbf{z}_t^e) + \mathbf{b}_d$. Next, we compute the error terms δ_t^d, δ_t^e as $\delta_t^d = \frac{\partial L(\tilde{\mathbf{d}}_t, d(e(\mathbf{d}_t)))}{\partial \mathbf{z}_t^d} = -(\frac{\tilde{\mathbf{d}}_t}{s_d(\mathbf{z}_t^d)} + \frac{1-\tilde{\mathbf{d}}_t}{1-s_d(\mathbf{z}_t^d)})s'_d(\mathbf{z}_t^d), \delta_t^e = (\mathbf{W}\delta_t^d + \delta(\rho/\tilde{\rho}_t + (1-\rho)/(1-\tilde{\rho}_t)))s'_e(\mathbf{z}_t^e)$. Finally, the gradient descents as partial derivatives can be computed as $\frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{W}} = \sum_{t=1}^T g_{dec}(\mathbf{x}_t) \delta_t^{dT} + \delta_t^e \mathbf{x}_t^T, \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{b}_e} = \sum_{t=1}^T \delta_t^e, \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{b}_d} = \sum_{t=1}^T \delta_t^d, \frac{\partial J(\mathbf{W}, \mathbf{b}_e, \mathbf{b}_d, \mathbf{W}', \mathbf{c})}{\partial \mathbf{c}} = \mathbf{c}^T - \sum_{t=1}^T \mathbf{x}_t$.

Update \mathbf{W} to satisfy orthogonality constraint: After

obtaining \mathbf{W} from SGD, we need to modify \mathbf{W} to satisfy the orthogonality constraint of $\mathbf{W}\mathbf{W}^T = \mathbf{I}$ in Eq. 5. This constraint is difficult to solve since it is non-convex and of high computational complexity. Our method to overcome this constraint follows the rigorous analysis in [62] by adopting a simple operation, $\mathbf{W} := (\mathbf{W}\mathbf{W}^T)^{-\frac{1}{2}}\mathbf{W}$, which sets the singular values of \mathbf{W} to be all ones. The above operation is conducted after every SGD step. The overall process for our deep learning based data minimization method is shown in Algorithm 3.

Model Stacking: Although Algorithm 3 is effective for solving Eq. 5, the learnt result heavily relies on the seeds, used to initialize the optimization process. Therefore, we use multiple hidden layers to stack the model in order to achieve more stable performance. In other words, our data minimization mechanism can also be used to build a deep network through model stacking. For the first layer in the deep learning model, we find the optimal layer by minimizing the objective function in Eq. 5. The representations learned by the first layer are then used as the input of the second layer, and so on so forth. Using a stacked deep auto-encoder, we can learn stable and finer-grained features to better represent the raw sensor data.